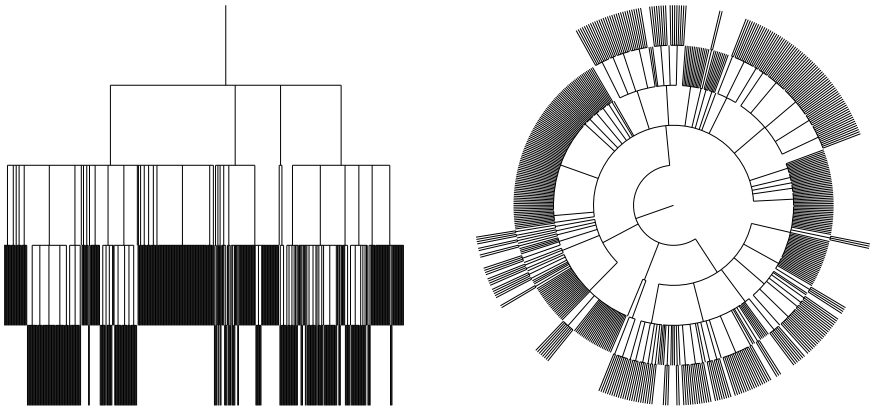
Projet de XML



Le but de ce projet est de convertir par programme, au format XML, un ensemble de données brutes décrivant la structure d'un arbre, puis d'extraire de ces données converties, à l'aide de XSLT, deux représentations graphiques de cet arbre au format SVG.

I) Conversion en XML d'un arbre au format CSV

a) Données brutes

La structure d'un arbre étiqueté par des chaînes de caractères peut être décrite dans le format CSV de la manière minimaliste suivante :

1. À chacun des nœuds de l'arbre est associé un unique identifiant entier.
2. Un premier fichier (*e.g.* `links.csv`) spécifie les liens de parenté entre les nœuds de l'arbre, et seulement ceux-ci : une ligne du fichier autre que la première contient l'entrée n, m si et seulement si le nœud d'identifiant m est l'un des fils du nœud d'identifiant n.
3. Un second fichier (*e.g.* `nodes.csv`) spécifie l'étiquette de chaque nœud : une ligne du fichier autre que la première commence par n, s si et seulement si le nœud d'identifiant n est étiqueté par la chaîne de caractères s. Ces deux valeurs peuvent éventuellement être suivies d'autres informations quelconques, inutiles à la description complète de l'arbre.

Deux fichiers CSV fournis avec cet énoncé, `treeoflife_links.csv` et `treeoflife_nodes.csv`, vous sont donnés à titre d'exemple : ils décrivent la structure d'un *arbre phylogénétique*. Les nœuds de cet arbre sont étiquetés par des noms de groupes d'êtres vivants, chaque nœud représentant le dernier ancêtre commun à tous ses descendants [1].

b) Conversion des données brutes

Votre premier travail sera de :

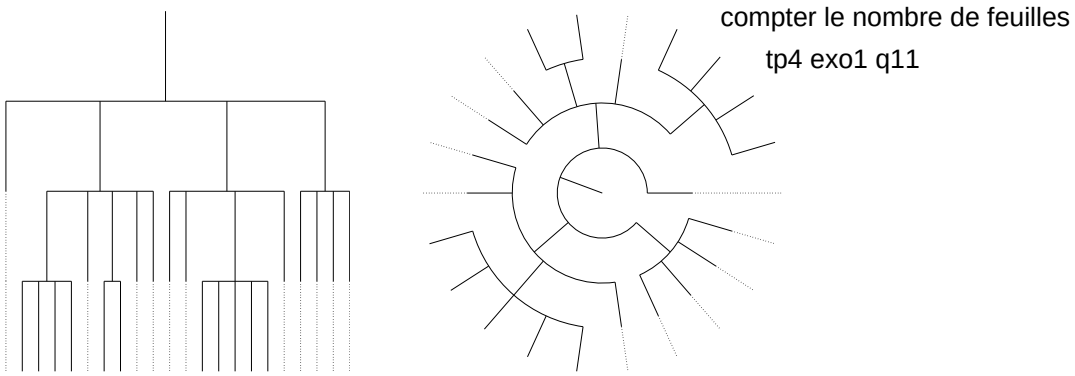
1. Choisir un format XML permettant de représenter des arbres étiquetés par des chaînes de caractères.
2. Écrire dans le langage de votre choix (C, Java, OCaml, Python, ...) un programme prenant en entrée les noms de deux fichiers CSV décrivant un arbre, et produisant un fichier XML dans le format choisi. Les informations autres que les liens de parenté des nœuds et leur étiquetage seront conservées dans le XML, mais simplement ignorées dans la production de la représentation graphique de l'arbre.
3. Écrire un Schéma XML (fichier `xsd`) validant votre format - là encore, en ignorant les informations non liées à la structure de l'arbre et son étiquetage.

1. Vous pouvez les ouvrir avec un éditeur de texte, mais il est plus facile de visualiser leurs contenus avec un tableur tel que `soffice` (le séparateur est la virgule).

II) Conversion des données XML en SVG

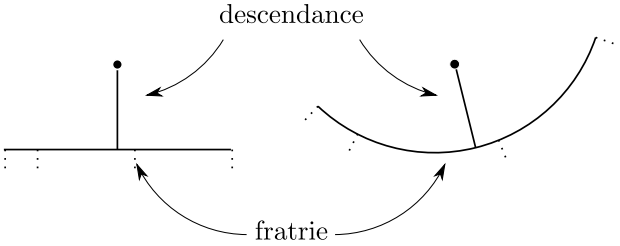
Votre second travail sera d'écrire un jeu de fichiers XSLT permettant de transformer tout arbre représenté dans le format XML choisi en deux représentations graphiques de cet arbre, toutes deux au format SVG : une représentation "rectangulaire", une autre "circulaire", décrites ci-dessous.

Une représentation rectangulaire doit occuper toute la largeur et la hauteur du document. Une représentation circulaire doit s'inscrire dans le plus grand cercle traçable dans le document, centré verticalement et horizontalement. On ne se préoccupera pas du rendu des étiquettes – cela peut être une des extensions de votre projet.



Cas des arbres complets. Un arbre est dit *complet* si toutes ses feuilles sont à la même profondeur. Voici la manière dont on représente un arbre complet.

Les nœuds de l'arbre sont ici identifiés avec des coordonnées dans le document SVG produit. Les liens de parenté entre les nœuds sont graphiquement représentés par deux sortes de tracés : des *tracés de descendance*, complétés par des *tracés de fratrie*.



1. De chaque nœud part un tracé de descendance, allant vers le bas du document dans la représentation rectangulaire, vers le cercle dans lequel s'inscrit le diagramme dans la représentation circulaire. Les longueurs de tous les tracés de descendance sont égales.
2. Le tracé de fratrie issu d'un nœud qui n'est pas une feuille va de son premier fils jusqu'à son dernier fils, en passant par tous ses autres fils. Ce tracé est un segment horizontal dans la représentation rectangulaire, un arc de cercle de même centre que le document dans la représentation circulaire.
3. Dans la représentation rectangulaire, les extrémités des tracés de descendance des feuilles sont disposées à espacements égaux sur toute la largeur du bord inférieur du document. Dans la représentation circulaire, ces extrémités sont disposées à espacements égaux sur le cercle dans lequel s'inscrit le diagramme.
4. Dans les deux représentations, l'extrémité du tracé de descendance d'un nœud qui n'est pas une feuille est exactement au milieu de son tracé de fratrie.

Cas des arbres quelconques. La *complétion* d'un arbre est obtenue de la manière suivante : pour chaque feuille de l'arbre qui n'est pas de profondeur maximale, on ajoute à l'arbre un unique chemin partant de cette feuille et atteignant la profondeur maximale de l'arbre.

Les chemins ajoutés à l'arbre représenté au début de cette section pour obtenir sa complétion sont tracés en pointillés. Les liens de parenté d'un arbre quelconque devront simplement être disposés aux mêmes emplacements que les liens de sa complétion déjà présents dans l'arbre l'initial – *comme si* les liens ajoutés étaient présents [2].

Remarques. Il est raisonnable et même probablement indispensable, pour construire les deux représentations d'un arbre, d'écrire *plusieurs* fichiers XSLT :

1. Un ou plusieurs fichiers XSLT produisant, à partir du XML initial, un ou plusieurs fichiers XML temporaires contenant des informations nécessaires à la construction des deux représentations, mais qui ne peuvent être calculées sans un premier parcours complet de l'arbre, *e.g.* pour chaque nœud, nombre de feuilles du sous-arbre associé, nombre de feuilles à gauche de ce sous-arbre, etc.
2. Deux fichiers XSLT, chacun construisant l'une des deux représentations à partir du dernier fichier XML temporaire produit.

Vous aurez par ailleurs besoin, pour la représentation circulaire, de fonctions trigonométriques. Elle sont utilisables en ajoutant à la balise `<stylesheet>` du XSLT l'attribut suivant :

```
xmlns:math="http://www.w3.org/2005/xpath-functions/math"
```

Ces fonctions peuvent ensuite s'écrire `math.cos(...)`, `math.sin(...)` (argument en `xsd:double` de résultat `xsd:double`), `math.pi()`, etc.

III) Modalités

a) Binômes

Le projet doit être impérativement réalisé en binôme. Toutefois, en cas d'impossiblité majeure de satisfaire cette contrainte (nombre impair d'étudiants, conditions d'études particulières, etc.), nous vous invitons à en discuter le plus tôt possible avec votre enseignant de cours magistral - aucun travail non réalisé en binôme ne sera accepté sans son accord explicite. La répartition des tâches au sein d'un binôme doit être raisonnablement équilibrée. En cas de déséquilibre avéré, les notes finales pourront être individualisées.

b) Individualité de chaque projet

De manière évidente, votre code doit être strictement personnel. Tout partage de code entre binômes est évidemment interdit [3]. Il relève de votre responsabilité de faire en sorte que votre code reste inaccessible aux autres binômes : par exemple, si vous vous servez d'un dépôt `git`, ce dépôt doit être privé.

c) Forme du rendu

Les dates de rendu et de soutenance seront précisées ultérieurement. Votre rendu consistera en :

– Tous les fichiers autres que les fichiers CSV de votre projet (le code de votre extracteur, les fichiers `xsd` et `xsl`).

2. À titre d'exemple, les deux premières figures de cet énoncé représentent l'arborescence du vivant : près de 36000 nœuds, environ 5 secondes de traitement de la compilation du parser jusqu'à la génération des deux SVG.

3. La soutenance de projet est un examen comme un autre. Le plagiat en projet constitue une fraude aux examens passible de lourdes sanctions discplinaires – ce cas s'est produit hélas plus d'une fois dans cette UFR.

- Un fichier texte nommé README contenant vos noms et numéros d'étudiants, et précisant la manière d'utiliser vos fichiers.
- Un rapport de quelques pages au format PDF décrivant votre projet, et expliquant et justifiant les choix de conception ou d'implémentation,

Tous ces éléments seront placés dans une unique archive compressée en `.tar.gz`. L'archive devra s'appeler *`nom1-nom2`*`.tar.gz`, et s'extraire dans un répertoire *`nom1-nom2`*`/`, où *`nom1`* et *`nom2`* sont les noms des deux personnes constituant le binôme. Par exemple, si vous vous appelez Denis Diderot et René Descartes, votre archive devra s'appeler `diderot-descartes.tar.gz` et s'extraire dans un répertoire `diderot-descartes/`.